WEST Search History

Hide Items Restore Clear Cancel

DATE: Wednesday, December 08, 2004

Hide?	Set Name	Query	Hit Count
		DB=USPT,EPAB,JPAB,DWPI,TDBD; PLUR=YES; OP=OR	
☐	L25	configur$9 with reset$4 with ((prior or before) near2 (initializ$9 or startup or start-up))	7
☐	L24	l17 and L23	18
☐	L23	L22 with default	96
☐	L22	(chang$4 or alter$4 or modif$9 or switch$4 or control$4) with configur$9 with (latch or flipflop or register)	10714
☐	L21	(chang$4 or alter$4 or modif$9 or switch$4 or control$4) near5 (configuration adj register) near5 default	3
☐	L20	(chang$4 or alter$4 or modif$9 or switch$4 or control$4) near5 (configuration adj register)	1497
☐	L19	l12 and L17	4
☐	L18	l4 and L17	9
☐	L17	l13 or l14 or l15 or L16	4858
☐	L16	326/38,40,46.ccls.	1724
☐	L15	710/10,104.ccls.	1093
☐	L14	716/16.ccls.	389
☐	L13	713/1,2,100.ccls.	2023
☐	L12	l3 same configur$9 same (flipflop or flip-flop or (flip adj flop) or latch or register)	35
☐	L11	l3 same (configuration adj register)	6
☐	L10	L9 same read$4	2
☐	L9	(load$4 near3 configur$9 near3 value) with (flipflop or flip-flop or (flip adj flop) or latch$3)	14
☐	L8	(pre-configur$9 with FPGA)	5
☐	L7	L3 near5 configur$4	89
☐	L6	pre-configur$4 near5 (prior or before) near5 initializ$9	2
☐	L5	L3 near5 (configur$4 or reconfigur$4)	91
☐	L4	L3 with (configur$4 or reconfigur$4)	112
☐	L3	((on-chip) or (single adj chip) or chip) adj device	5051
☐	L2	((on-chip) or (single adj chip) or chip) near2 device	36282
☐	L1	((on-chip) or (single adj chip) or chip) near2 (device or system)	49110

L6: Entry 1 of 2 File: USPT Sep 12, 2000

DOCUMENT-IDENTIFIER: US 6119192 A
TITLE: Circuit and method for configuring a bus bridge using parameters from a supplemental parameter memory

Abstract Text (1):
A circuit and method are provided for initializing configuration registers in a bus bridge controller so that peripheral devices can acquire functionality prior to initialization by a normal system Basic Input/Output System (BIOS) routine. The initialization information is stored in a supplemental configuration Electronically Erasable Programmable Read Only Memory (EEPROM), separate from a conventional system BIOS ROM. In addition, the register configuration information is stored in a flexible format so that only certain devices are pre-configured, or that certain devices are initialized before others. The invention has particular usefulness in personal computing systems which include a Peripheral Component Interconnect (PCI) bus, an ISA bus, and a system management bus (SMB) for providing access to the EEPROM.

Detailed Description Text (10):
In a preferred embodiment, Supplemental Configuration Data Memory 160 is a serial E.sup.2 PROM which is programmed with data in the manner depicted in FIG. 5. As seen there, every fifth addressed byte, beginning with address 0, contains an index to one of PCI Configuration Registers 210. These values, of course, can be programmed in any manner desired, so that is possible to configure such register set in any desired (not just sequential) sequence. In addition, the register configuration information may include information so that only certain devices are pre-configured, or that certain devices are initialized before others. As noted before, a configuration routine "stop" value of AA (hex) is indicated at location 5n. Since memory 160 is conveniently placed on SMB 156, updates are easily implemented, and initialization is accomplished without burdening other busses within computing system 100.

Previous Doc Next Doc Go to Doc#

L10: Entry 1 of 2 File: USPT Dec 17, 2002

DOCUMENT-IDENTIFIER: US 6496971 B1
TITLE: Supporting multiple FPGA configuration modes using dedicated on-chip processor

Detailed Description Text (15):
Each row of configuration memory cells has an associated Y address line. Line 105 in FIG. 3 is the Y address line associated with the row of configuration memory cell 100. Each column has an associated X address line. Line 106 in FIG. 3 is the X address line associated with the column of configuration memory cell 100. If a digital high is present on the X address line of a configuration memory cell, and a digital high is present on the Y address line of the configuration memory cell, and a global enable line 107 is pulsed with a high pulse, then an AND gate 108 of the configuration memory cell outputs a digital high pulse thereby enabling an access transistor 109. If a frame buffer 110 drives a digital value onto a data line 111 of the configuration memory cell, then this digital value will be loaded into the configuration memory cell and will be stored by the latch. If frame buffer 110 is not output enabled, then latch 101 drives the stored configuration data through enabled access transistor 109 and onto data line 111. This data can then be read by processor 9 via a readback buffer 112.

L8: Entry 1 of 5 File: USPT Apr 1, 2003

DOCUMENT-IDENTIFIER: US 6542844 B1
TITLE: Method and apparatus for tracing hardware states using dynamically reconfigurable test circuits

Detailed Description Text (6):
Referring now to FIG. 3, details of PCI interface 26A and color plane interface 27A of FIG. 2 are depicted. PCI interface FPGA 31 is configured at startup by Configuration ROM 32. The power-up configuration of PCI interface FPGA is pre-configured to read FPGA configuration information severally from configuration ROM 32. Once the configuration contained within configuration ROM 32 is loaded, PCI interface FPGA implements a PCI bus interface and may be further configured via this interface.

L8: Entry 2 of 5 File: USPT Apr 30, 2002

DOCUMENT-IDENTIFIER: US 6381732 B1
TITLE: FPGA customizable to accept selected macros

Brief Summary Text (8):
FPGA 110 also includes dedicated configuration logic circuits to program the user logic circuits. Specifically, each CLB, IOB, PSM, and PIP contains a configuration memory (not shown) which must be configured before each CLB, IOB, PSM, or PIP can perform a specified function. Some FPGAs may also include pre-configured logic circuits that are configured by the manufacturer to perform specific functions.

Brief Summary Text (9):
For example, some FPGAs are pre-configured to include a PCI bus interface. Typically the configuration memories within an FPGA use static random access memory (SRAM) cells. The configuration memories of FPGA 110 are connected by a configuration structure (not shown) to configuration port 120 through a configuration access port (CAP) 125. A configuration port (a set of pins used during the configuration process) provides an interface for external configuration devices to program the FPGA. The configuration memories are typically arranged in rows and columns. The columns are loaded from a frame register which is in turn sequentially loaded from one or more sequential bitstreams. (The frame register is part of the configuration structure referenced above.) In FPGA 110, configuration access port 125 is essentially a bus that couples configuration port 120 to the configuration structure of FPGA 110.

Previous Doc Next Doc Go to Doc#

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#
End of Result Set

☐ Generate Collection Print

L8: Entry 5 of 5 File: USPT Dec 12, 2000

DOCUMENT-IDENTIFIER: US 6160418 A
TITLE: Integrated circuit with selectively disabled logic blocks

Detailed Description Text (6):
FIG. 5 is a simplified schematic diagram of an FPGA 500 in accordance with another embodiment of the present invention. The programmable interconnect, IOB, and PADs of FPGA 500 are omitted in FIG. 5 for clarity. FPGA 500 comprises an array of CLBs, a disable circuit 570, and disable signals 571-573. FPGA 500 can be pre-configured with various built-in functions that may not all be needed simultaneously. For example, CLB group 550, which comprises CLBs 541-543 and 551-553, may be pre-configured to be a PCI interface; while CLB group 504, which comprises CLBs 545, 554, and 555, may be pre-configured to be an SBUS interface. Thus, FPGA 500 can be used in systems using either PCI bus or SBUS. However, the manufacturer of FPGA 500 may be able to price an SBUS version of FPGA 500 differently from a PCI bus version. Thus, the manufacturer of FPGA 500 would desire to create two separate product lines with FPGA 500. Therefore, portions of FPGA 500 can be selectively disabled to create several product lines from FPGA 500.

Detailed Description Text (7):
Specifically, disable circuit 570 can disable CLB group 550 by driving disable line 573 to a disable logic state. Similarly, disable circuit 570 can disable CLB group 504 by driving disable line 572 to a disable logic state. Furthermore, disable circuit 570 can disable a CLB group 506, comprising CLBs 514, 515, 524, 525, 534, 535, and 544, by driving disable line 571 to the disable logic state. An FPGA that includes on-board memory for pre-configuration is described by Lawman in U.S. patent application Ser. No. 09/000,519, entitled "DECODER STRUCTURE AND METHOD FOR FPGA CONFIGURATION" by Gary R. Lawman.

Detailed Description Text (8):
FIG. 6 shows a CLB 600 for an FPGA in accordance with another embodiment of the present invention. CLB 600 comprises a CLB disable circuit 610 that can be programmed after manufacturing of the FPGA to either disable or enable CLB 600. Thus, in an FPGA using CLB 600 in place of standard CLBs, each CLB can be individually enabled or disabled after manufacturing of the FPGA. Thus, the FPGA manufacturer need only manufacture a single type of FPGA to be able to offer multiple families of FPGAs having varying CLB matrix sizes as well as different pre-configured functionality.

Previous Doc Next Doc Go to Doc#

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#

Generate Collection Print

L11: Entry 4 of 6 File: USPT Apr 25, 1989

DOCUMENT-IDENTIFIER: US 4825438 A
TITLE: Bus error detection employing parity verification

Detailed Description Text (10):
The Versatile Bus allows great flexibility for configuring interfaces that meet the exact needs of the system designer without requiring a redesign of the interface logic on the chips being interconnected. A Versatile Bus is standardized with respect to voltage levels and clock speeds, but it allows the system designer to select data widths, arbitration schemes, function codes, transaction overlap, latencies and acknowledge formats, that best suit his design requirement. No hardware redesign of the VLSIC devices he is employing is required. A Versatile Bus achieves this flexibility through a single design which can be configured for the applications requirements of diverse systems and devices, so that any of several different interconnect types may be accomplished using the same set of input/output pins. The system designer may set a Versatile Bus to his own requirements by setting a configuration register within each interconnected chip device. The configuration register may be set before the device is even soldered into a machine system, or it may be set and reset through an optional device maintenance interface. This maintenance and initialization interface, called the VM Node, is a separate interconnection to the Versatile Interface Logics from the bus itself. Ths VM Node maintenance interface also supports off-line test and on-line error correction/reconfiguration in response to fault conditions recognized during Versatile Bus operation.

Previous Doc Next Doc Go to Doc#

L9: Entry 1 of 14 File: USPT Nov 30, 2004

DOCUMENT-IDENTIFIER: US 6826732 B2
TITLE: Method, system and program product for utilizing a configuration database to configure a hardware digital system

Detailed Description Text (16):
As discussed above, a digital design, whether realized utilizing physical integrated circuitry or as a software model such as simulation model 300, typically includes configuration latches utilized to configure the digital design for proper operation. In contrast to prior art design methodologies, which employ stand-alone configuration software created after a design is realized to load values into the configuration latches, the present invention introduces a configuration specification language that permits a digital designer to specify configuration values for signals as a natural part of the design process. In particular, the configuration specification language of the present invention permits a design configuration to be specified utilizing statements either embedded in one or more HDL files specifying the digital design (as illustrated in FIG. 4A) or in one or more external configuration files referenced by the one or more HDL files specifying the digital design (as depicted in FIG. 4B).

Detailed Description Text (23):
As illustrated, LDial 500, like all Dials, logically has a single input 502, one or more outputs 504, and a mapping table 503 that maps each input value to a respective associated output value for each output 504. That is, mapping table 503 specifies a one-to-one mapping between each of one or more unique input values and a respective associated unique output value. Because the function of an LDial is to specify the legal values of configuration latches, each output 504 of LDial 500 logically controls the value loaded into a respective configuration latch 505. To prevent conflicting configurations, each configuration latch 505 is directly specified by one and only one Dial of any type that is capable of setting the configuration latch 505.

Detailed Description Text (28):
FIG. 5B illustrates another important property of LDials (and other Dials that directly specify configuration latches). In particular, as shown diagrammatically in FIG. 5B, designers, who are accustomed to specifying signals in HDL files, are permitted in a configuration specification statement to specify signal states set by a Dial rather than values to be loaded into an "upstream" configuration latch that determines the signal state. Thus, in specifying LDial 506, the designer can specify possible signal states for a signal 514 set by a configuration latch 512. Similarly, in specifying LDial 510, the designer can specify possible signal states for signal 522 set by configuration latch 520. The ability to specify signal states rather than latch values not only coincides with designers' customary manner of thinking about a digital design, but also reduces possible errors introduced by the presence of inverters between the configuration latch 512, 520 and the signal of interest 514, 522, as discussed further below.

Detailed Description Text (41):
Referring now to FIG. 6A, there is depicted a diagrammatic representation of an Integer Dial ("IDial") in accordance with a preferred embodiment of the present invention. Like an LDial, an IDial directly specifies the value loaded into each of

one or more configuration latches 605 by indicating within mapping table 603 a correspondence between each input value received at an input 602 and an output value for each output 604. However, unlike LDials, which can only receive as legal input values the enumerated input values explicitly set forth in their mapping tables 503, the legal input value set of an IDial includes all possible integer values within the bit size of output 604. (Input integer values containing fewer bits than the bit size of output(s) 604 are right justified and extended with zeros to fill all available bits.). Because it would be inconvenient and tedious to enumerate all of the possible integer input values in mapping table 603, mapping table 603 simply indicates the manner in which the integer input value received at input 602 is applied to the one or more outputs 604.

Detailed Description Text (71):
As described above, the configuration specification language of the present invention advantageously permits the specification of the output values of LDials and IDials by reference to signal names (e.g., "sig1"). As noted above, a key motivation for this feature is that designers tend to think in terms of configuring operative signals to particular signal states, rather than configuring the associated configuration latches. In practice, however, a signal that a designer desires to configure to a particular state may not be directly connected to the output of an associated configuration latch. Instead, a signal to be configured may be coupled to an associated configuration latch through one or more intermediate circuit elements, such as buffers and inverters. Rather than burdening the designer with manually tracing back each configurable signal to an associated configuration latch and then determining an appropriate value for the configuration latch, configuration compiler 808 automatically traces back a specified signal to the first storage element (i.e., configuration latch) coupled to the signal and performs any necessary inversions of the designer-specified signal state value to obtain the proper value to load into the configuration latch.

Detailed Description Text (72):
With reference now to FIG. 9A, there is illustrated a portion of a digital design including an LDial 900 that controls the states of a plurality of signals 904a-904e within the digital design. When configuration compiler 808 performs a traceback of signal 904a, no inversion of the designer-specified signal states is required because signal 904a is directly connected to configuration latch 902a. Accordingly, configuration compiler 808 stores into configuration database 814 the designer-specified values from the configuration specification statement of LDial 900 as the values to be loaded into configuration latch 902a. Traceback of signal 904b to configuration latch 902b similarly does not result in the inversion of any designer-specified values from the configuration specification statement of LDial 900 because the only intervening element between signal 904b and configuration register 902b is a non-inverting buffer 906.

Previous Doc Next Doc Go to Doc#

L9: Entry 3 of 14 File: USPT Dec 17, 2002

DOCUMENT-IDENTIFIER: US 6496971 B1
TITLE: Supporting multiple FPGA configuration modes using dedicated on-chip processor

Detailed Description Text (15):
Each row of configuration memory cells has an associated Y address line. Line 105 in FIG. 3 is the Y address line associated with the row of configuration memory cell 100. Each column has an associated X address line. Line 106 in FIG. 3 is the X address line associated with the column of configuration memory cell 100. If a digital high is present on the X address line of a configuration memory cell, and a digital high is present on the Y address line of the configuration memory cell, and a global enable line 107 is pulsed with a high pulse, then an AND gate 108 of the configuration memory cell outputs a digital high pulse thereby enabling an access transistor 109. If a frame buffer 110 drives a digital value onto a data line 111 of the configuration memory cell, then this digital value will be loaded into the configuration memory cell and will be stored by the latch. If frame buffer 110 is not output enabled, then latch 101 drives the stored configuration data through enabled access transistor 109 and onto data line 111. This data can then be read by processor 9 via a readback buffer 112.

L21: Entry 2 of 3 File: USPT Jun 16, 1998

DOCUMENT-IDENTIFIER: US 5768584 A
TITLE: ROM chip enable encoding method and computer system employing the same

CLAIMS:

16. The memory controller as recited in claim 15 wherein said configuration register contains a default value upon power-up of said memory controller.

Previous Doc Next Doc Go to Doc#

First Hit Fwd Refs Previous Doc Next Doc Go to Doc#

Generate Collection Print

L9: Entry 12 of 14 File: USPT Apr 25, 2000

DOCUMENT-IDENTIFIER: US 6054871 A
TITLE: Method for self-reconfiguration of logic in a field programmable gate array

Detailed Description Text (42):
The reprogramming of function generators 101-102 is controlled by circuitry (not shown) located outside of CLB 100, typically by other CLBs of the FPGA. This circuitry determines when the function generators 101-102 are to provide different output functions and then initiates the reprogramming. The reprogramming of function generators 101-102 is performed as previously described. That is, the function generators are reconfigured from the ROM look up table configuration to the user RAM configuration, data values representative of the new function are written to the latches of the user RAM, and the function generators are reconfigured from the user RAM configuration to the ROM look up table configuration, thereby loading the data values representative of the new function in the ROM look up table. In the foregoing manner, function generators 101-102 are able to implement more complex logic functions. The cost of implementing these more complex logic functions is the additional time required to reconfigure the function generators 101-102 each time the function generators 101-102 are to perform different functions. Although the present invention has been described in connection with the implementation of a 4-to-1 multiplexer, it is understood that other logic functions can be implemented by function generators 101-102 in accordance with the principles of the present invention. For example, logic functions, such as multiplication by a constant which must be modified occasionally, can be implemented by reconfiguring the function generators in the various CLBs.

Previous Doc Next Doc Go to Doc#

L24: Entry 1 of 18 File: USPT Nov 9, 2004

DOCUMENT-IDENTIFIER: US 6816918 B2
TITLE: Flexible apparatus for setting configurations using an EEPROM

Abstract Text (1):
A method for flexibly configuring default values of a network device through an EEPROM interface is disclosed. A header is received from an EEPROM through the EEPROM interface and it is determined from the header whether any default value of the network device should be updated, and if any, how many should be updated. At least one configuration instruction is fetched from the EEPROM when it is determined that the network device should be updated. The at least one configuration instruction is interpreted and a register default value of the default values corresponding to the interpreted at least one configuration instruction is changed.

Brief Summary Text (9):
When the RESET signal goes to in-active, network switch/hub chip start to fetch data from external EEPROM automatically. Most of the network switch/hub chips will fetch data from EEPROM address 00h (the first entry), and fetch the other data in sequence. In order to change some register default values or set chip configuration, the chip vendor will provide a register set (a part of chip register file) which are downloadable from EEPROM. Each entry of EEPROM is pre-defined and will directly map to one (or some) entry of register set inside network switch/hub chip as described in FIG. 2.

Brief Summary Text (13):
It is an object of this invention to overcome the drawbacks of the above-described conventional network devices and methods. The present invention provides for a new approach for chip vendors to provide system integrators a dynamic configuration using low cost EEPROM. With this approach, system integrators will have flexibility to change the default values of all configure-able registers inside a network device, such as a switch/hub chip.

Brief Summary Text (14):
According to one aspect of this invention, a method for flexibly configuring default values of a network device through an EEPROM interface. A header is received from an EEPROM through the EEPROM interface and it is determined from the header whether any default value of the network device should be updated. At least one configuration instruction is fetched from the EEPROM when it is determined that the network device should be updated. The at least one configuration instruction is interpreted and a register default value of the default values corresponding to the interpreted at least one configuration instruction is changed.

Detailed Description Text (3):
To achieve this flexible configuration apparatus, network switch/hub chip vendor should build-in a circuit (called Configuration Instruction Interpreter, CII) inside the chip to interpret configuration instruction. When RESET signal goes to in-active, the CII of network switch/hub chip start to fetch header (the first entry) from external EEPROM automatically, then the key is obtained. If the key value is not matched with the magic number pre-defined inside network switch/hub chip, it indicates that it is not necessary to change any chip default value, and

download sequence might be skipped. While key is match, CII continuously fetches configuration instruction from EEPROM, and changes the corresponding (defined in address index of configuration instruction) register default value to the desired value by interpreting instruction. This process will be repeated until all instruction download completely. Additionally, since the number of default values needing to be updated is determined from the start, the time needed to perform the updated is less than the equivalent updating performed in the prior art methods and systems.

Current US Cross Reference Classification (4):
710/10

Current US Cross Reference Classification (5):
710/104

Current US Cross Reference Classification (7):
713/1

Current US Cross Reference Classification (8):
713/100

Current US Cross Reference Classification (9):
713/2

CLAIMS:

1. A method for flexibly configuring default values of a network device through an EEPROM interface, comprising: receiving a header from an EEPROM through the EEPROM interface where the header provides information about a total number of configuration instructions in the EEPROM; determining from the header whether any default value of the network device should be updated; fetching at least one configuration instruction from the EEPROM when the determining step determines that the network device should be updated; interpreting said at least one configuration instruction; and changing a register default value of said default values corresponding to said interpreted at least one configuration instruction.

Previous Doc Next Doc Go to Doc#

L24: Entry 14 of 18 File: USPT Mar 9, 1999

DOCUMENT-IDENTIFIER: US 5881281 A
TITLE: Method and apparatus for automatically loading configuration data on reset into a host adapter integrated circuit

Brief Summary Text (15):
In some embodiments, a comparator determines whether the configuration data in the shift register has a predetermined value. A default configuration value is initially loaded into a register which is accessible from the host. If the configuration data in the shift register has a predetermined value (for example, "OFF"), then the register accessible from the host is not changed, thereby leaving the register loaded with the default configuration value. If, on the other hand, the configuration data does have the predetermined value, then the register is loaded with the configuration data in the shift register. In some embodiments, a pullup or pulldown resistor is coupled to the data input terminal of the host adapter integrated circuit so that the predetermined value (for example, "FF") will be shifted into the shift register in the event that no external device is coupled to the data input terminal. In some embodiments, which of two default configuration values is loaded into the externally accessible register is determined by a digital value received from another input terminal of the host adapter integrated circuit.

Current US Original Classification (1):
713/1

L24: Entry 17 of 18 File: USPT Mar 24, 1998

DOCUMENT-IDENTIFIER: US 5732281 A
**** See image for Certificate of Correction ****
TITLE: Programmable power management circuit for a power supply in a computer system

Detailed Description Text (8):
Power control state machine 16 senses power from power supply 30 when computer 10 is turned on by a user. Computer 10 is preferably turned on when the user engages an on/off switch. Power control state machine 16 also monitors for voltages on any auxiliary power supplies and for voltage on a CMOS battery within computer 10. After it senses battery-backed power, power control state machine 16 causes reset generator 18 to reset configuration registers 22 with hardware default configuration data.

Detailed Description Text (11):
Reset generator 18 causes the hardware default configuration data to be loaded into configuration registers 22 in response to a RESET signal from power control state machine 16.

Detailed Description Text (30):
In step 54, power control state machine 16 causes reset generator 18 to reset configuration registers 22 with the hardware default configuration data.

Current US Cross Reference Classification (1):
710/10